University of Science and Technology
Faculty of Computer Science and Information Technology

Information Security Master Program

Course Title: Cloud Computing Security

Lecture (2): IT Infrastructure Security

Reference: Cloud Computing Security and Privacy, by Tim Mather, Subra and Lattif, -------Chapter (3)

Instructor: Prof. Noureldien Abdelrahman

This lecture describes:

- IT infrastructure security at the network, host, and application levels.

IT infrastructure security is an approach that used to study the security issues of an organization in three levels, network, host and application. Information security practitioners commonly use this approach; to investigate security issues, such as threats, attacks, and countermeasures.

In this lecture and next lectures we will discuss cloud security using the infrastructure security approach in the context of SPI service delivery models (SaaS, PaaS, and IaaS) and the cloud business model (public, private, and hybrid clouds).

When discussing public clouds the scope of infrastructure security is limited to the layers of infrastructure that move beyond the organization's control and into the hands of service providers (i.e., when responsibility to a secure infrastructure is transferred to the cloud service provider or CSP, based on the SPI delivery model).

2.1 Infrastructure Security: The Network Level

When looking at the network level of infrastructure security, it is important to distinguish between public clouds and private clouds. With private clouds, there are no new attacks, vulnerabilities, or threats that information security personnel need to consider because there is changes in *the network topology*. Although the organization's IT architecture may change with the implementation of a private cloud, the current network topology will probably not change significantly.

2.1.1 Network Level Security Requirements

With public clouds, the changing of security requirements will require changes to the network topology. In this case there are four security requirements:

• Ensuring the confidentiality and integrity of the organization's data-in-transit to and from the public cloud provider.
• Ensuring proper access control (authentication, authorization, and auditing) to whatever resources used at public cloud provider.
• Ensuring the availability of the Internet-facing resources in a public cloud that are being used by the organization, or have been assigned to the organization by the public cloud providers
• Replacing the established model of network zones and tiers with domains.

1- Ensuring Data Confidentiality and Integrity

When using clouds some resources and data that are previously confined (shut in) to a private network are now exposed to the Internet, and to a shared public network belonging to a third-party cloud provider.

In web applications the use of HTTPS (instead of HTTP) would have mitigated the integrity risk, users not using HTTPS (but using HTTP) or using other applications did face an increased risk that their data could have been altered in transit without their knowledge.

2- Ensuring Proper Access Control

Since some subset of these resources (or maybe even all of them) is now exposed to the Internet, an organization using a public cloud faces a significant increase in risk to its data. The ability to control access and audit the operations of your organization's data at the cloud is a genuine security concern. Also you will have decreased access to relevant network-level logs and data, and a limited ability to thoroughly conduct investigations and gather forensic data.

3- Ensuring the Availability of Internet-Facing Resources

When using clouds an increased amount of data and an increased number of organizational applications are hosted in the cloud, this requires to ensure the availability of cloud-provided resources.

4- Replacing the Established Model of Network Zones and Tiers with Domains

The established isolation model of network zones and tiers no longer exists in the public IaaS and PaaS clouds. For years, network security has relied on zones, such as intranet versus extranet or development systems from production systems, to segregate network traffic for improved security. This model was based on exclusion—only individuals and systems in specific roles have access to specific zones. Similarly, systems within a specific tier often have only specific access within or across a specific tier.+

The traditional model of network zones and tiers has been replaced in public cloud computing with "security groups," "security domains"or"virtual data centers" that have logical separation between tiers but are less precise and afford less protection than the formerly established model.

In the established model of network zones and tiers, not only were development systems logically separated from production systems at the network level, but these two groups of systems were also physically separated at the host level (i.e., they ran on physically separated servers in logically separated network zones). With cloud computing, however, this separation no longer exists. The cloud computing model of separation by domains provides logical separation for addressing purposes only. There is no longer any "required" physical separation, as a development domain and a production domain may very well be on the same physical server.

2.1.2 Network-Level Mitigation

Given the factors discussed in the preceding sections, what can you do to mitigate these increased risk factors?

First, note that network-level risks exist regardless of what aspects of "cloud computing" services are being used (e.g., software-as-a-service, platform-as-a-service, or infrastructure-as-a-service). The primary determination of risk level is therefore not which *aaS is being used, but rather whether the organization intends to use or is using a public, private, or hybrid cloud.

If the organization is large enough to afford the resources of a private cloud, the risks will decrease—assuming having a true private cloud that is internal to the network. In some cases, a private cloud located at a cloud provider's facility can help meet the security requirements but will depend on the provider capabilities and maturity.

If the organization has to use public cloud, the risks to confidentiality can be reduced by using encryption; specifically by using validated implementations of cryptography for data-in-transit. Also secure digital signatures make it much more difficult, if not impossible, for someone to tamper with your data, and this ensures data integrity.

Availability problems at the network level are far more difficult to mitigate with cloud computing—unless the organization is using a private cloud that is internal to the network topology. Even if the private cloud is a private (i.e., non-shared) external network at a cloud provider's facility, you will face increased risk at the network level. A public cloud faces even greater risk.

Table 3-1 lists security controls at the network level.

TABLE 3-1. Security controls at the network level

Threat outlook	**Low (with the exception of DoS attacks)**
Preventive controls	Network access control supplied by provider (e.g., firewall), encryption of data in transit (e.g., SSL, IPSec)
Detective controls	Provider-managed aggregation of security event logs (security incident and event management, or SIEM), network-based intrusion detection system/intrusion prevention system (IDS/IPS)

2.2 Infrastructure Security: The Host Level

Although there are no known new threats to hosts that are specific to cloud computing, some virtualization security threats—such as VM escape, system configuration drift, and insider threats due to the weak access controls to the hypervisor—carry into the public cloud computing environment.

The dynamic nature (elasticity) of cloud computing can bring new operational challenges from a security management perspective. The operational model motivates rapid provisioning and fleeting instances of VMs. Managing vulnerabilities and patches is therefore much harder than just running a scan, as the rate of change is much higher than in a traditional data center.

2.2.1 SaaS and PaaS Host Security

In general, CSPs do not publicly share information related to their host platforms, host operating systems, and the processes that are in place to secure the hosts, since hackers can exploit that information when they are trying to intrude into the cloud service. Hence, in the context of SaaS (e.g., Salesforce.com, Workday.com) or PaaS (e.g., Google App Engine, Salesforce.com's Force.com) cloud services, host security is obscure/unclear to customers and the responsibility of securing the hosts is relegated to the CSP.

To get assurance from the CSP on the security hygiene of its hosts, customers should ask the vendor to share information under a nondisclosure agreement (NDA) or simply demand that the CSP share the information via a controls assessment framework such as SysTrust or ISO 27002. From a controls assurance perspective, the CSP has to ensure that appropriate preventive and detective

controls are in place and will have to ensure the same via a third-party assessment or ISO 27002 type assessment framework.

In summary, host security responsibilities in SaaS and PaaS services are transferred to the CSP. However, a customer, still own the risk of managing information hosted in the cloud services. It's the customer responsibility to get the appropriate level of assurance regarding how the CSP manages host security hygiene.

2.2.2 IaaS Host Security

Given that almost all IaaS services available today employ virtualization at the host layer, host security in IaaS is directly defined by virtualization security.

2.2.2.1 Virtualization Security

1- Introduction

Virtualization provides techniques for using resources and devices without considering their position and physical layout. It supports an encapsulated environment, guaranteeing machines isolation, hardware independence and hardware partitioning.

In general, virtualization solutions allow different users to manage and share physical hardware by supporting multiple shared environments that are isolated, while running on the same infrastructure.

Virtualization introduces many benefits that strengthen cloud flexibility and efficiency, summarized as follows:

- Server consolidation and reduced costs for system operation and management
- Optimized resource utilization
- Responding to application and user requirements dynamically at runtime, allowing users to share resources.
- Multiple execution environments, where users can select the environment that best suits their requirements in a given timeframe.

- Simplified management through a single view of all (distributed) resources, supporting interoperability between heterogeneous systems and centralized control of the environment.

2- Virtualization Components

Virtualization techniques and virtualized architectures introduce an additional layer of execution, including their own administrator role (virtualization admin), which require proper management and security protection.

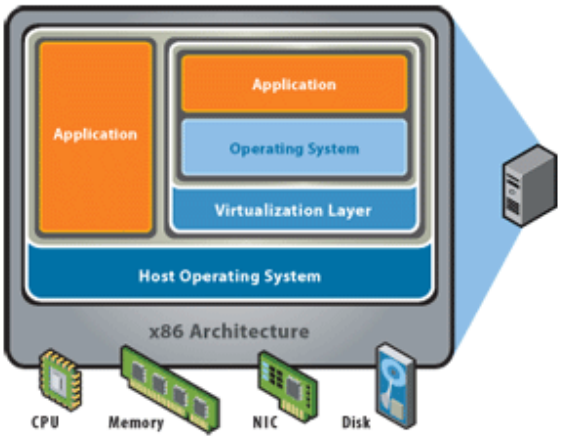
This layer is made up of several different components, each with a role in the virtualization process, each representing a potential new target for malicious attacks. This section discusses some of these components.

- **The hypervisor**

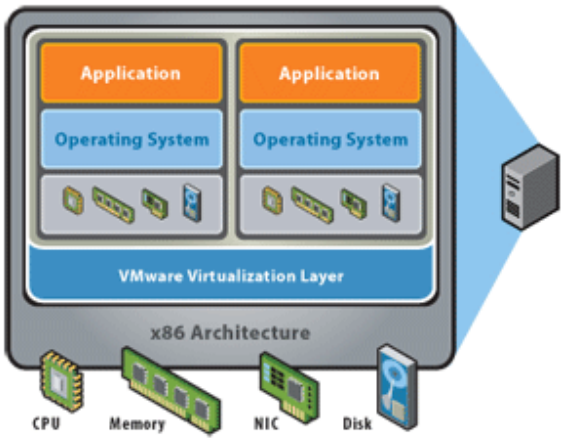
The hypervisor is the component that acts as a mediator between virtual machines and the underlying physical devices. It mediates all hardware requests by the virtual machines down to the physical hardware, sharing physical devices as resources.

It implements the virtual machine monitor (VMM) which is an application component of the hypervisor that keeps track of activities carried out by virtual machines (i.e., it manages VM applications), forwards hardware request to physical resources, provides replicated platforms, and supports resource sharing between different virtual machines. It has the responsibility to guarantee end users virtualization transparency.

Hypervisors can be of two types, bare-metal or hosted.



Host Virtualization



Bare Meal Virtualization

Hosted virtualization software runs as an application or "guest" on top of a general-purpose operating system. In this architecture, a base operating system (such as Windows) is first installed. A piece of software called a hypervisor or virtual machine monitor (VMM) is installed on top of the host OS, and allows users to run various guest operating systems within their own application windows

Hosted Hypervisor relies on an underlying OS, more security implications are there because of the reliance on the underlying OS, and this technology is used in VMware Workstation/Server /Player and MS Virtual PC.

Bare-metal virtualization interfaces directly with computer hardware, without the need for a host operating system. In this architecture, the hypervisor is installed that communicates directly with system hardware rather than relying on a host operating system. This technology is used in VMware ESX Server, XEN and MS Hyper V.

- **Guest machines**

Guest machines are also known as **virtual machines**, instantiate the virtualized (encapsulated) system made of the operating system and applications, using the hardware abstraction provided by the virtual machine monitor. Guest machines are isolated by the hypervisor, which controls their activities, and behave as if they were in a single execution environment with their own dedicated resources.

Each guest machine can install a different operating system to support virtualization heterogeneity. A special, guest-machine is the container which is an operating system-level virtualization in which the kernel of an operating system allows the existence of multiple isolated user-space instances.

- **The host machine**

This is the real physical machine and its operating system (**host operating system**) that hosts the virtualized environment. The host operating system directly manages the physical hardware underlying the virtualized environment and is where the hypervisor runs.

- **The management server**

The management server is the virtualization platform made up of a set of components for directly managing the virtual machines, consolidating services, allocating resources, migrating virtual machines, and assuring high availability.

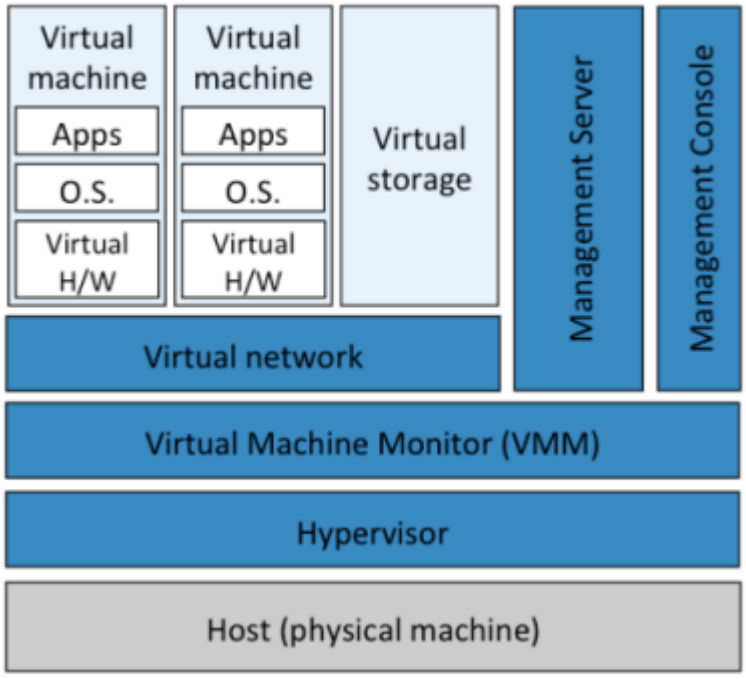
- **The management console**

This is the component that provides access to a management interface to the virtualization product for configuring and managing virtual machines. Virtual machines can thus be created, added, modified, deleted or configured. The management console can be provided as a standalone client or via a web interface to visually handle management server functionalities. Examples of management consoles include the VMware vSphere client console and the VMware vSphere web client.

- **Virtualized storage**

The virtualized storage abstracts all physical storage components in a single storage device that can be accessed either over the network or through a direct connection. Storage virtualization introduces additional management overhead, due to the fact that stored data can be only logically partitioned in different storage locations while belonging to the same shared storage.

Figure 1-1 Components of Virtualization



3- Virtualization technology classification

Virtualization technologies are classified as:

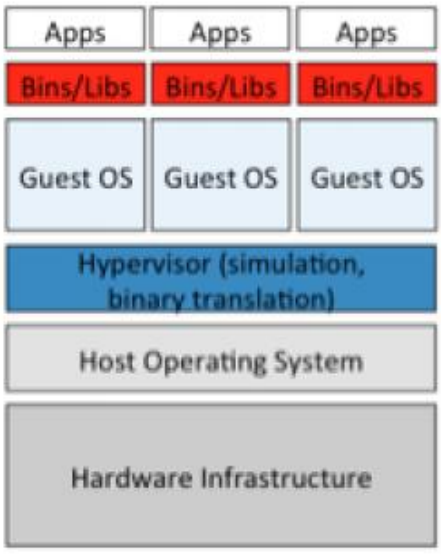
- **Full hardware virtualization** supports the virtualization of (x86) systems by simulating the underlying hardware. The hardware is simulated in software by each virtual machine. The guest OS is completely separated from the underlying hardware, access to which is mediated by the virtualization layer (virtual machine monitor). Full virtualization provides a solution with highest isolation and security, while it decreases performance and adds more overhead. Examples of solutions supporting hardware virtualization include VirtualBox, Virtual PC, VMware, Win4Lin, Xen, and User Mode Linux.

- **Operating-system-level virtualization** is based on an operating system that supports multiple instances of isolated user-space**, called containers**. Each container can target a single application and install only the needed software and libraries to run this application. The host machine's hardware resources are partitioned between different guest machines. The host OS deploys many instances of guest OSes, with a lightweight execution of the operating system or application.

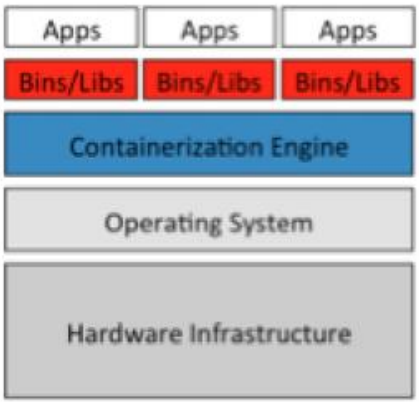
Resources are assigned to containers that represent a set of processes, files, and

partitions. This approach provides high performance, low overhead, and permits the execution of the same OS as the host machine. Examples of solutions supporting operating system-level virtualization include Docker, Virtuozzo, OpenVZ, and Solaris Containers.

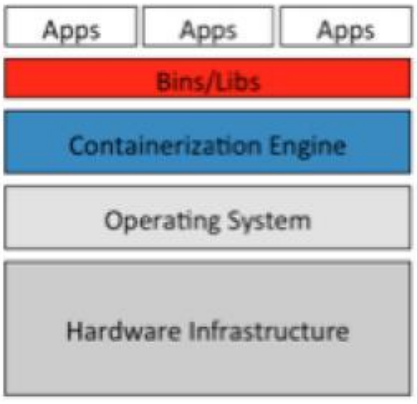
- **Application-level virtualization** increases programs' portability between different software and hardware architectures. It is based on various components, including: a portable language, a compiler between source code and an architecture-independent representation (bytecode), a bytecode interpreter, and an execution environment that translates bytecode into low-level operations on the host machine. Examples of solutions supporting application-level virtualization include Java VM, Microsoft .NET, Perl, Python, and Ruby.



(a) Full virtualization



(d) OS-level virtualization



(e) Application-level virtualization

4. Categorization of threats to virtualization

Virtualization shares with traditional computing environments a number of security issues that affect operating systems, communication protocols, and applications. In some cases, such issues are even exacerbated (more badly) by the presence of virtualized components, requiring special care to address them.

On the other hand, virtualization also introduces a number of virtualization-specific security issues that require ad hoc solutions. For instance, new security issues introduced by virtualization involve:

i) Multitenancy allowing cross-platform information flow between customers who share the same physical host,

ii) Adversaries able to execute arbitrary out-of-the-guest code without owning the required access rights, and

iii) Special security requirements in virtualized storage for keeping data secure during cloud events such as migration.

5. Threat taxonomy

To address the security concerns introduced by a virtual environment, **a threat model is needed to classify all possible attacks.** In the following, we present a general consequence-based categorization of threats as defined in IETF RFC 2828.

(Unauthorized) Disclosure: gaining unauthorized access to protected information

In virtual environments, where physical resources are shared between tenants, there may be a set of behaviors that result in the disclosure of sensitive information. For instance, *exposure* via hunt/search in virtualized environments is even more serious than in physical systems. While *interception* is a common threat in physical systems (e.g., networking environments), its effect is further exacerbated in virtual environments because it permits cross-inspection of various tenant's data flow, as well as topology inference that could serve to set up a denial-of-service attack.

Deception: intentionally attempting to mislead other entities.

An unauthorized entity gains access to a system or performs a malicious act by posing as an authorized entity via:

i) **masquerading** (i.e., spoofing, identity fraud),
ii) **falsifying data to deceive or mislead an authorized entity** (e.g., wiretapping, reply attacks),
iii) **falsely denying responsibility for an act** (i.e., repudiation),
iv) **misleading authorized users to perform actions that, alone or combined with the actions of other users, will have negative consequences** on the system (e.g. social engineering attacks).

Deception is a common threat for physical systems, but in the case of virtualized environments, identity handling may be more difficult.

Disruption: causing failure or degradation of systems, negatively affecting the services they provide.

This threat may occur by

i) **directly *harming* system components or communication channels**,
ii) ***alternating or corrupting* system operations by adversely modifying system functions to ind**uce delivery of corrupted data,
iii) **interrupting delivery of system services**.

Usurpation: an unauthorized entity that may gain control over a system.

Usurpation refers to *misappropriation* of identity via service, functionality, or data theft, or *misuse* of action (e.g., tampering), which causes a system component to

perform a function or service that is detrimental to system security. For example, by sending a message that appears to have been originated from a privileged entity, an attacker may raise its own privilege level to carry out a privilege escalation attack.

In virtualized environments, privilege rise can be even more dangerous than in a physical environment because of multitenancy and the hierarchical structure of administrator privileges. In addition, VMM is a crucial target for usurpation-based misappropriation, due to its role in virtualization, as well as to the presence of vulnerabilities that allow guest-OS users the potential to execute arbitrary code on the host OS.

6. Vulnerabilities in virtual environments

Vulnerabilities are exists in almost all the virtualization components.

a- Guest OS and Host OS Vulnerabilities

The guest and the host OSs share many weaknesses and vulnerabilities. For this reason, we have grouped them into the same section. Both guest and host OS are affected *by* ***operating-system-level vulnerabilities***, which are considered very critical both because of the large number of vulnerabilities and because of their potential impact if administrator access is gained at the operating-system level.

As with all virtualization components, the sharing of resources is one of the main sources of vulnerability, along with the use of monitoring or acting tools. At the VM level, vulnerabilities that lead to DoS often target the VM itself or the functionalities it provides.

The guest OS is the starting point for many attacks. Exploiting vulnerability in the guest or host OS may **expose the system to the following consequences**.

- *Attack the VM or other VMs (direct attack)*: an attacker can take advantage of relaxed access control or intentional inter-VM communications. This attack depends on host configuration and access control.

- ***Virtual machine escape*** *attack: This is* an exploit in which the attacker runs code on a **VM** that allows an operating system running within it to break out and interact directly with the hypervisor. Such an exploit could give the attacker

access to the host operating system and all other virtual machines (**VMs**) running on that host.

- *Attacking the hypervisor*: this kind of attack usually starts in a guest OS and is hypervisor-dependent.
- *Attack the hardware on the host*: hardware platforms often request firmware updates. By accessing this mechanism from a VM, an attacker could upload firmware to favor the attack. For this reason, many hypervisors filter such commands if possible.

b- Containers

In a container-based scenario, an application's operating environment is virtualized. The result is an isolated container in which the application can run. Containers are different from guest OS and virtual machines in general, but they share some security vulnerabilities, such as access control, privilege permissions, and coding weaknesses.

Specifically, container attacks mainly aim to escape from container isolation, similar to VM escape.

Exploiting vulnerability in the container can expose to the following consequences.

- *Attack the application*: this mainly leads to disclosure of data.
- *Attack the whole container*: this can lead to denial of service and disruption of the services provided.

- *Attack the host architecture*: this could lead to arbitrary file modification on the host system and execution of code (disruption).

C- Hypervisor / VMM / Management server and console

The hypervisor, the VMM and their management tools share weaknesses and vulnerabilities. For this reason, we grouped them into the same section. When not specifically mentioned, we refer to both hypervisor, VMM, and the corresponding management server and console as hypervisor.

From a security perspective, the ideal hypervisor is a system with the fewest possible lines of code. In other words, a hypervisor delivers all its basic

functionalities (managing hardware for guest VMs, maintaining a secure environment and isolation between the VMs) with no need for extra services such as plug-n-play, wireless services or graphical user interface, this will increase its security and reducing the number of exploitable vulnerabilities.

However, the hypervisor is a fundamental component in any virtualized system and is thus the most attractive target for attack. As a result, it shows a large number of vulnerabilities, mainly related to its fundamental security features:

i) VM isolation, which is the target of the majority of attackers,
ii) internal software-based channels for communication with VM, which are suitable intrusion channels for any attacker, and
iii) VMM or additional API for inspection, which are vulnerable like any software-system entry point.

The effects of these vulnerabilities include DoS that could corrupt hypervisor memory space and possibly execute arbitrary code, obtain sensitive information from hypervisor stack content or gain privileged access to the hypervisor. Hypervisors also show vulnerabilities when providing specific functionality for VM management, such as VM migration

Exploiting vulnerabilities in the hypervisor/VMM/management server and console can expose systems to the following consequences.

- *Attack on the VMs hosted by the hypervisor*: this can lead either to disclosure of information from the VMs or full/partial disruption of system availability.

- *Attack on the host platform*: this mainly leads to denial of service (disruption). This kind of problem is sometimes due to issues related to hardware and firmware. Attackers inside any guest VM could get data and code by digging into the host-operating-system memory space.

2.3 Infrastructure Security: The Application Level

Application or software security should be a critical element of your security program. The application security spectrum ranges from standalone single-user applications to sophisticated multiuser e-commerce applications used by millions of users. Web applications such as content management systems (CMSs), wikis, portals, bulletin boards, and discussion forums are used by small and large organizations. A large number of organizations also develop and maintain custom-built web applications for their businesses using various web frameworks (PHP,† .NET,‡ J2EE,§ Ruby on Rails, Python, etc.).

In this section, we will limit our discussion to web application security: web applications in the cloud accessed by users with standard Internet browsers, such as Firefox, Internet Explorer, or Safari, from any computer connected to the Internet.

Since the browser has emerged as the end user client for accessing in-cloud applications, it is important for application security programs to include browser security into the scope of application security. Together they determine the strength of end-to-end cloud security that helps protect the confidentiality, integrity, and availability of the information processed by cloud services.

2.3.1 Application-Level Security Threats

According to SANS (*SysAdmin, Audit, Network and Security*, www.sans.org) , web application vulnerabilities in open source as well as custom-built applications accounted for almost half the total number of vulnerabilities discovered. **The existing threats include:** cross-site scripting (XSS), SQL injection, malicious file execution, and other threats resulting from programming errors and design flaws.

Armed with knowledge and tools, hackers are constantly scanning web applications (accessible from the Internet) for application vulnerabilities. They are then exploit the vulnerabilities they discover for various illegal activities including financial fraud, intellectual property theft, converting trusted websites into malicious servers serving client-side exploits, and phishing scams.

It has been a common practice to use a combination of perimeter security controls and network- and host-based access controls to protect web applications deployed

in a tightly **controlled environment, including corporate intranets and private clouds, from external hackers.**

Web applications deployed in a public cloud (the SPI model) must be designed for an Internet threat model, and security must be embedded into the Software Development Life Cycle (SDLC).

Additionally, you should be cognizant/aware of application-level DoS and DDoS attacks that can potentially disrupt cloud services. DoS attacks on pay-as-you-go cloud applications will result in a dramatic increase in your cloud utility bill: you'll see increased use of network bandwidth, CPU, and storage consumption. This type of attack is also being characterized as *economic denial of sustainability* (EDoS).

2.3.1.1 End User Security

Customers of a cloud service, are responsible for end user security tasks—security procedures to protect your Internet-connected PC—and for practicing "safe surfing." Protection measures include use of security software, such as anti-malware, antivirus, personal firewalls, security patches, and IPS-type software on your Internet-connected computer.

Browsers have become the ubiquitous "operating systems" for consuming cloud services. All Internet browsers routinely suffer from software vulnerabilities that make them vulnerable to end user security attacks. Hence, our recommendation is that cloud customers take appropriate steps to protect browsers from attacks. To achieve end-to-end security in a cloud, it is essential for customers to maintain good browser hygiene. The means keeping the browser (e.g., Internet Explorer, Firefox, Safari) patched and updated to mitigate threats related to browser vulnerabilities.

2.3.1.2Who Is Responsible for Web Application Security in the Cloud?

Depending on the cloud services delivery model (SPI) and service-level agreement (SLA), the scope of security responsibilities will fall on the shoulders of both the customer and the cloud provider. The key is to understand what your security

responsibilities are versus those of the CSP. In that context, recent security surveys have highlighted the fact that lack of transparency in security controls and practices employed by CSPs is a barrier to cloud adoption.

Due to this lack of transparency, customers are left with no choice but to trust their CSPs to disclose any new vulnerability that may affect the confidentiality, integrity, or availability of their application.

The following sections discuss the web application security in the context of the SPI cloud service delivery model.

2.3.3 SaaS Application Security

The SaaS model dictates that the provider manages the entire suite of applications delivered to users. Therefore, SaaS providers are largely responsible for securing the applications and components they offer to customers.

Customers are usually responsible for **operational security functions, including user and access management as** supported by the provider.

It is a common practice for prospective customers, to request information related to the provider's security practices. This information should encompass design, architecture, development, black- and white-box application security testing, and release management.

Some customers go to the extent of hiring independent security vendors to perform penetration testing of SaaS applications to gain assurance independently. However, penetration testing can be costly and not all providers agree to this type of verification.

Extra attention needs to be paid to the authentication and access control features offered by SaaS CSPs to their customers.

Usually that is the only security control available to manage risk to information. Most services, including those from Salesforce.com and Google, offer a web-based administration user interface tool to manage authentication and access control of the application.

Cloud customers should try to understand cloud-specific access control mechanisms—including support for strong authentication and privilege management based on user roles and functions—and take the steps necessary to protect information hosted in the cloud.

Table 3-3 lists security controls at the application level.

TABLE 3-3. Security controls at the application level

Threat outlook	**Medium**
Preventive controls	Identity management, access control assessment, browser hardened with latest patches, multifactor authentication via delegated authentication, endpoint security measures including antivirus and IPS
Detective controls	Login history and available reports from SaaS vendors

3.3.4 PaaS Application Security

By definition, a PaaS cloud (public or private) offers an integrated environment to design, develop, test, deploy, and support custom applications developed in the language the platform supports. PaaS application security encompasses two software layers:

- **Security of the PaaS platform itself** (i.e., runtime engine)
- **Security of customer applications deployed on a PaaS platform**

Generally speaking, PaaS CSPs (e.g., Google, Microsoft, and Force.com) are responsible for securing the platform software stack that includes the runtime engine that runs the customer applications. Since PaaS applications may use third-party applications, components, or web services, the third-party application provider may be responsible for securing their services.

Hence, customers should understand the dependency of their application on all services and assess risks pertaining to third-party service providers.

In the multitenant PaaS service delivery model, the core security tenets/principles are **containment** and **isolation of multitenant applications** from each other. In that model, access to your data should be restricted to your enterprise users and to applications that you own and manage. The **security model of the PaaS platform**

runtime engine is the CSP's intellectual property, and it is essential to delivering the "sandbox" architecture in a multitenant computing model.

*In general, a **sandbox** is an isolated computing environment in which a program or file can be executed without affecting the application in which it runs. Sandboxes are used by software developers to test new programming code*.

Hence, the sandbox characteristic of the platform runtime engine is central in maintaining the confidentiality and integrity of your application deployed in the PaaS. CSPs are responsible for monitoring new bugs and vulnerabilities that may be used to exploit the PaaS platform and break out of the sandbox architecture.

2.3.4.1 Customer-Deployed Application Security

PaaS developers need to get familiar with specific APIs to deploy and manage software modules that enforce security controls. Furthermore, given that the API is unique to a PaaS cloud service, developers are required to become familiar with platform-specific security features—available to them in the form of security objects and web services for configuring authentication and authorization controls within the application.

Developers should expect CSPs to offer a set of security features, including user authentication, single sign-on (SSO) using federation, authorization (privilege management), and SSL or TLS support, made available via the API.

2.3.5 IaaS Application Security

IaaS cloud providers (e.g., Amazon EC2, GoGrid, and Joyent) treat the applications on customer **virtual instances as a black box,** and therefore are completely **agnostic/** to the operations and management of the customer's applications.

The entire stack—customer applications, runtime application platform (Java, .NET, PHP, Ruby on Rails, etc.), and so on— runs on the customer's virtual servers and is deployed and managed by customers. To that end, **customers have full responsibility for securing their applications deployed in the IaaS cloud**.

Hence, customers should not expect any application security assistance from CSPs other than basic guidance and features related to firewall policy that may affect the

application's communications with other applications, users, or services within or outside the cloud.

Customers of IaaS clouds are responsible for all aspects of their application security and should take the steps necessary to protect their application to address application-level threats in a multitenant and hostile Internet environment. Table 3-5 lists security controls applicable to IaaS applications.

TABLE 3-5. Security controls applicable to IaaS applications

Threat outlook	**High**
Preventive controls	Application developed using security-embedded SDLC process, least-privileged configuration, timely patching of application, user authentication, access control, account management, browser hardened with latest patches, endpoint security measures including antivirus, IPS, host-based IDS, host firewall, and virtual private network (VPN) for administration
Detective controls	Logging, event correlation, application vulnerability scanning and monitoring